T-79.5501 Cryptology

Lecture 9 (Nov 15, 2005):

- Wiener's Low decryption Exponent Attack, Sec 5.7.3
- Security of the Rabin Cryptosystem, Sec 5.8.1
- Bleichenbacher's attack
- OAEP (Cryptosystem 5.4)
- Rabin OT
- 1-out-of-2 oblivious transfer

RSA Cryptosystem

$n = pq$ where p and q are two different large primes

$$\phi(n) = (p\text{-}1)(q\text{-}1)$$

a decryption exponent (private)

b encryption exponent (public)

$$ab \equiv 1 \left(\bmod \phi(n)\right)$$

RSA operation:

$$(m^b)^a \equiv m \pmod{n}$$

for all m, $0 \leq m < n$.

Wiener's result: It is insecure to select a shorter than about ¼ of the length of n.

RSA Equation

$$ab - k\,\phi(n) = 1$$

for some k where only b is known.

Additional information: $pq = n$ is known and $q < p < 2q$

$$n > \phi(n) = (p-1)(q-1) = pq - p - q + 1 \geq n - 3\sqrt{n}$$

Also we know that a, $b < \phi(n)$, hence $k < a$.

Wiener (1989) showed how to exploit this information to solve for a and all other parameters k, p and q, if a is sufficiently small.

Wiener's method is based on continued fractions.

Continued Fractions

Every rational number t has a unique representation as a finite chain of fractions

$$q_1 + \cfrac{1}{q_2 + \cfrac{1}{q_3 + \cfrac{1}{\ddots\ q_{m-1} + \cfrac{1}{q_m}}}}$$

and we denote $t = [q_1\, q_2\, q_3\ \ldots\ q_{m\text{-}1}\, q_m]$. The rational number $t_j = [q_1\, q_2\, q_3\ \ldots\ q_j]$ is called the j^{th} convergent of t. For $t = u/v$, just run the Euclidean algorithm to find the q_i, $i = 1, 2,\ldots,$m.

Convergent Lemma

Theorem 5.14 *Suppose that* gcd(u,v)= gcd (c,d) = 1 *and*

$$\left|\frac{u}{v} - \frac{c}{d}\right| < \frac{1}{2d^2}.$$

Then c/d is one of the convergents of the continued fraction expansion of u/v.

Recall the RSA problem: $ab - k\phi(n) = 1$

Write it as:

$$\frac{b}{\phi(n)} - \frac{k}{a} = \frac{1}{a\phi(n)}$$

Then, if $2a < \phi(n)$, then k/a is a convergent of $b/\phi(n)$.

Wiener's Theorem

If in RSA cryptosystem

$$a < \frac{1}{3}\sqrt[4]{n},$$

that is, the length of the private exponent a is less than about one forth of the length of n, then a can be computed in polynomial time with respect to the length of n.

Proof. First we show that *k/a* can be computed as a convergent of *b/n,* based on Euclidean algorithm, which is polynomial time. To see this, we estimate:

$$\left|\frac{b}{n}-\frac{k}{a}\right| = \left|\frac{ab-kn}{an}\right| = \left|\frac{1+k\phi(n)-kn}{an}\right| \le \frac{3k}{a\sqrt{n}} < \frac{3}{\sqrt{n}} < \frac{1}{2a^2}.$$

Wiener's Algorithm

Then the convergents $c_j/d_j = [q_1 \, q_2 \, q_3 \; ... \; q_j]$ of b/n are computed. For the correct convergent $k/a = c_j/d_j$ we have

$$bd_j - c_j \, \phi(n) = 1.$$

For each convergent one computes

$$n' = (d_j b - 1) / c_j$$

and checks if $n' = \phi(n)$. Note that $p + q = n - \phi(n) + 1$. Then if $n' = \phi(n)$, the equation

$$x^2 - (n - n' + 1)x + n = 0$$

has two positive integer solutions p and q.

PKCS#1

PKCS#1 v 1.5 before it was corrected:

$$EB = 00 \,||\, BT \,||\, PS \,||\, 00 \,||\, B$$

BT block type: 00, 01, tai 02.
(In public key encryption $BT = 02$)

The leftmost 00 guarantees that the plaintext after conversion to an integer is less than the RSA module n.

PKCS#1 v 1.5

Bleichenbacherin hyökkäys:

- Bob näkee salatun C jonka haluaa tulkita: $M = C^d \bmod n$
- Bob valitsee kokonaislukuja S ja laskee $C' = CS^e \bmod n$ ja lähettää tulokset C' Alicelle.
- Alice laskee $(C')^d \bmod n = MS \bmod n$ ja ilmoittaa Bobille onko tulos laillinen, siis PKCS standardin mukainen, vai ei.
- Jos C' on laillinen, niin Bob tietää että luvun $MS \bmod n$ kaksi ensimmäistä tavua ovat 00 || 02
- Silloin Bob saa tietää että seuraava epäyhtälö pätee:

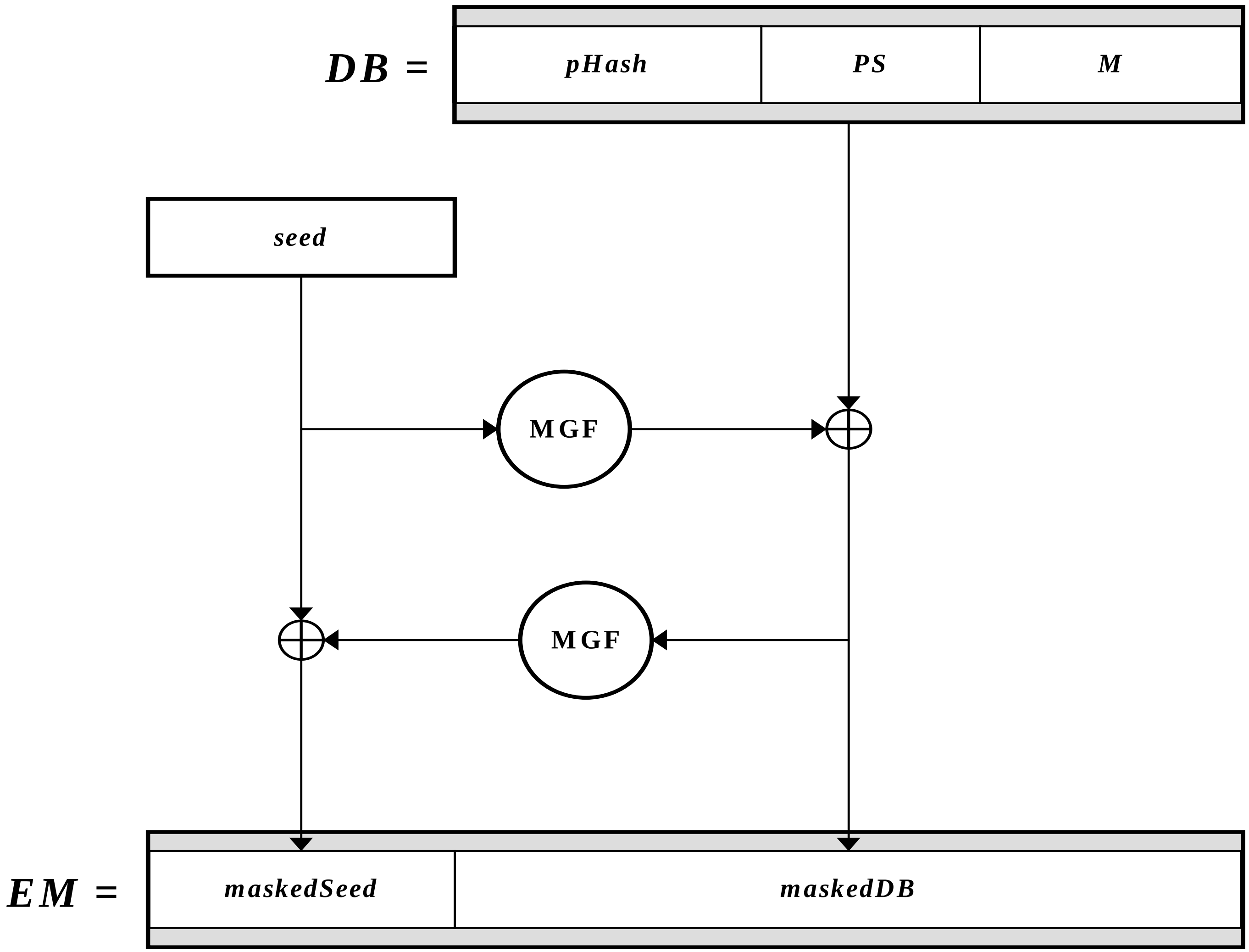
$$2B \leq MS \bmod n < 3B$$

missä $B = 2^{8(k-2)}$ ja k on RSA-moduulin n pituus tavuina.

- Keräämällä useita ($\sim 2^{20}$) epäyhtälöitä Bob voi määrittää M:n

PKCS#1 v 2.1 EME-OAEP

Based on Bellare and Rogaway's Optimal Asymmetric Encryption scheme (1994)



Rabin OT

Two players: sender (Alice) and receiver (Bob)

Goal: Alice has one bit. Bob is allowed to try once to get the bit. His success probability is ½ . Alice does not know, if Bob gets the bit or not.

Protocol:

1. Alice sets up an RSA cryptosystem: p, q, n, a, b, with $ab \equiv 1 \bmod \Phi(n)$.
2. Alice encrypts the bit s, to get $c = \{encode(s)\}^b \bmod n$, and sends c, b and n to Bob.
3. Bob selects x, $0 < x < n$, at random, computes $y = x^2 \bmod n$, and sends y to Alice.
4. Alice finds the four square roots of y and picks one, say z, of them and sends it to Bob.
5. If $z \neq \pm x \bmod n$, Bob can factor n, compute $a = b^{-1} \bmod \Phi(n)$, and decrypt c, with probability ½. Alice does not know if $z \neq \pm x \bmod n$.

1-out-of-2 OT using RSA

Protocol:

Two players: sender (Alice) and receiver (Bob)

Goal: Alice has two secret bits. Bob is allowed two see exactly one of them. Alice does not know, which of the two bits Bob gets.

Alice's inputs: two bits a_0 and a_1

Bob's input: one bit s

Protocol: OT(a_0, a_1; s)

Output to Alice: nothing

Output to Bob: $a_s = (s \oplus 1)\ a_0 \oplus s\ a_1$

Next we see how to implement OT(a_0, a_1; s) assuming Bob is honest, which is the case of "private information retrieval".

1-out-of-2 Oblivious Transfer

Protocol:

1. Alice sets up an RSA cryptosystem Alice sets up an RSA cryptosystem: p, q, n, a, b, with $ab \equiv 1 \bmod \phi(n)$, and sends n and b to Bob.

Hard-core bit for the RSA function: For randomly chosen x, given y, n, b, where $y = x^b \bmod n$ finding the lsb of x is essentially as hard as finding all of x (see also Stinson, Section 5.9)

2. Bob selects a random m with lsb r_s and computes the ciphertext $c_s = m^b \bmod n$. Bob selects c_{1-s} at random, and sends c_s and c_{1-s} , that is, c_0 and c_1 to Alice.
3. Alice decrypts c_0 and c_1 and gets the lsb:s r_0 and r_1 of the plaintexts. She then conceals the bits a_0 and a_1 by computing $a'_0 = r_0 + a_0 \pmod 2$ and $a'_1 = r_1 + a_1 \bmod 2$, and sends a'_0 and a'_1 to Bob.
4. Bob then gets a_s from a'_s as he knows r_s. Alice does not know s.